

CLAIMS**What is Claimed:**

1. A method for processing a data stream embodying a hierarchically structured document, said method comprising:

partitioning said data stream into fixed length segments utilizing said hierarchical structure to determine a length of each fixed length segment; and

processing said fixed length segments in a pipeline fashion.

2. A method in accordance with claim 1, further comprising converting said hierarchically structured document into said data stream.
3. A method in accordance with claim 2, wherein said act of converting comprises converting said hierarchically structured document formatted in a transport protocol into said data stream.
4. A method in accordance with claim 3, wherein said transport protocol comprises at least one of a hypertext transport protocol (HTTP) and a file transport protocol (FTP).
5. A method in accordance with claim 1, wherein said act of processing comprises decoding said fixed length segments.
6. A method in accordance with claim 5, wherein said act of decoding comprises multipurpose mail extensions (MIME) decoding.
7. A method in accordance with claim 5, further comprising parsing said decoded fixed length segments.
8. A method in accordance with claim 7, wherein act of parsing comprises extensible markup language (XML) parsing.
9. A method in accordance with claim 7, further comprising:

partitioning said parsed fixed length segments into fragments; and

storing said fragments in a storage medium, wherein a size of a fragment is determined in accordance with characteristics of said storage medium.

10. A method in accordance with claim 9, wherein said storage medium comprises a database.

11. A method in accordance with claim 10, further comprising:

creating a database table comprising:

meta data associated with said document;

queries over said document and respective results;

a first fragment of said document; and

locations within said database and sizes of all fragments of said document other than said first fragment; and

storing said database table in said database.

12. A method in accordance with claim 11, further comprising:

retrieving said database table from said database; and

utilizing said retrieved database table to retrieve all fragments of said document

13. A method in accordance with claim 9, said act of processing comprising:

retrieving said fragments from said storage medium; and

serializing said retrieved fragments into fixed length segments.

14. A method in accordance with claim 9, further comprising encoding said serialized fixed length segments.

15. A method in accordance with claim 14, wherein said act of encoding comprises multipurpose mail extensions (MIME) encoding.

16. A method in accordance with claim 14, further comprising converting said encoded fixed length segments into a transport protocol.
17. A method in accordance with claim 16, said transport protocol comprises at least one of a hypertext transport protocol (HTTP) and a file transport protocol (FTP).
18. A method in accordance with claim 1, further comprising:
 - receiving a signal indicative of at least one query over said document, wherein said data stream comprises tokens indicative of a structure of said hierarchically structured document;
 - evaluating said tokens in the passing data stream in response to said at least one query;
 - identifying fixed length segments of said data stream to be utilized to satisfy each query; and
 - individually processing each fixed length segment to satisfy said at least one query.
19. A method in accordance 18, wherein:
 - said data stream comprises a sequence of nodes;
 - a node is one of a container node and a value node;
 - each token is indicative of at least one of a container node and a value node;
 - a value node comprises a value associated with said document; and
 - a container node comprises at least one of:
 - attributes of said document; and
 - an ordered set of at least one of a container node and a value node.
20. A method in accordance with claim 18, further comprising:

comparing a current token of a fixed length segment, as it is received, to every query; and

determining if any of the at least one query is satisfied by the current token.

21. A method in accordance with claim 20, further comprising:

if a particular query is satisfied by particular token, updating the particular token in accordance with the particular query.

22. A method in accordance with claim 18, wherein:

said query is formatted in an XPath-type language; and

said document if formatted in an XML-type language.

23. A method in accordance with claim 1, further comprising:

determining if a selected read query is in a read cache, wherein:

a read query is utilized to read a fixed length segment of passing stream data;

if said selected read query is in said read cache, indicating if said selected query is satisfied;

if said selected read query is not in said read cache, add said selected read query to a prefetch set;

obtaining at least one fixed length segment of passing stream data; and

consuming said at least one fixed length segment of passing stream data until all unresolved queries in said prefetch set are resolved.

24. A method in accordance with claim 1, further comprising:

creating a new read cache comprising existing unresolved queries and receiving a source data stream, wherein a read cache is utilized to read a fixed length segment of passing stream data;

creating an empty write cache, wherein a write cache is utilized to provide fixed length segments of stream data;

associating said empty write cache with an existing instance of a re-writer, wherein a re-writer is utilized to provide a copy of a fixed length segment of stream data; and

providing a copy of said source data stream as an output data stream from said empty write cache.

25. A method in accordance with claim 1, further comprising:

obtaining a fixed length segment of source stream data from an associated read cache;

consuming said fixed length segment of source stream data;

resolving pending queries on said source stream data;

creating a mutator stream of data from said source stream of data, wherein mutations in said mutator stream are in accordance with write requests cached in an associated write cache;

consuming an output stream of data from said mutator stream of data; and

resolving queries on said output stream;

discarding resolved queries; and

embedding values associated with said resolved queries in said output stream.

26. A method in accordance with claim 24, further comprising:

determining if a read-query to read said data stream is safe, wherein a safe read-query is a semantically correct query;

if said read-query is safe:

searching a write cache for a value associated with said read-query;

if said value is found in said write cache, returning said value to a user;

if said value is not found in said write cache, forwarding said read-query to a read cache;

if said query is not safe:

obtaining said output stream from a current instance; and

extracting said read-query from said obtained output stream.

27. A method in accordance with claim 24, further comprising:

determining if a write request to write a data stream is safe, wherein a safe write request is semantically correct;

if said write request is safe:

searching a write cache for a value associated with said write request;

if said value is found in said write cache, replacing said associated value with a value of said write request;

if said value is not found in said write cache, creating a new entry in said write cache having said value of said write request;

if said write request is not safe:

creating an intermediate stream writer for separating existing write requests from said unsafe write request;

creating a new stream writer; and

returning said new stream writer to a user.

28. A computer-readable medium having computer-executable instructions for processing a data stream embodying a hierarchically structured document by performing acts comprising:

partitioning said data stream into fixed length segments utilizing said hierarchical structure to determine a length of each fixed length segment; and

processing said fixed length segments in a pipeline fashion.

29. A computer-readable medium in accordance with claim 28, further comprising converting said hierarchically structured document into said data stream.

30. A computer-readable medium in accordance with claim 29, wherein said act of converting comprises converting said hierarchically structured document formatted in a transport protocol into said data stream.

31. A computer-readable medium in accordance with claim 30, wherein said transport protocol comprises at least one of a hypertext transport protocol (HTTP) and a file transport protocol (FTP):

32. A computer-readable medium in accordance with claim 28, wherein said act of processing comprises decoding said fixed length segments.

33. A computer-readable medium in accordance with claim 32, wherein said act of decoding comprises multipurpose mail extensions (MIME) decoding.

34. A computer-readable medium in accordance with claim 32, further comprising parsing said decoded fixed length segments.

35. A computer-readable medium in accordance with claim 34, wherein act of parsing comprises extensible markup language (XML) parsing.

36. A computer-readable medium in accordance with claim 35, further comprising:

partitioning said parsed fixed length segments into fragments; and

storing said fragments in a storage medium, wherein a size of a fragment is determined in accordance with characteristics of said storage medium.

37. A computer-readable medium in accordance with claim 36, wherein said storage medium comprises a database.

38. A computer-readable medium in accordance with claim 37, further comprising:

creating a database table comprising:

meta data associated with said document;

queries over said document and respective results;

a first fragment of said document; and

locations within said database and sizes of all fragments of said document other than said first segment; and

storing said database table in said database.

39. A computer-readable medium in accordance with claim 38, further comprising:

retrieving said database table from said database; and

utilizing said retrieved database table to retrieve all fragments of said document

40. A computer-readable medium in accordance with claim 36, said act of processing comprising:

retrieving said fragments from said storage medium; and

serializing said retrieved fragments into fixed length segments.

41. A computer-readable medium in accordance with claim 36, further comprising encoding said serialized fixed length segments.

42. A computer-readable medium in accordance with claim 41, wherein said act of encoding comprises multipurpose mail extensions (MIME) encoding.

43. A computer-readable medium in accordance with claim 41, further comprising converting said encoded fixed length segments into a transport protocol.

44. A computer-readable medium in accordance with claim 43, said transport protocol comprises at least one of a hypertext transport protocol (HTTP) and a file transport protocol (FTP).
45. A computer-readable medium in accordance with claim 28, further comprising:
- receiving a signal indicative of at least one query over said document, wherein said data stream comprises tokens indicative of a structure of said hierarchically structured document;
 - evaluating said tokens in the passing data stream in response to said at least one query;
 - identifying fixed length segments of said data stream to be utilized to satisfy each query; and
 - individually processing each fixed length segment to satisfy said at least one query.
46. A computer-readable medium in accordance with claim 45, wherein:
- said data stream comprises a sequence of nodes;
 - a node is one of a container node and a value node;
 - each token is indicative of at least one of a container node and a value node;
 - a value node comprises a value associated with said document; and
 - a container node comprises at least one of:
 - attributes of said document; and
 - an ordered set of at least one of a container node and a value node.
47. A computer-readable medium in accordance with claim 45, further comprising:
- comparing a current token of a fixed length segment, as it is received, to every query; and

- determining if any of the at least one query is satisfied by the current token.
48. A computer-readable medium in accordance with claim 47, further comprising:
if a particular query is satisfied by particular token, updating the particular token in accordance with the particular query.
49. A computer-readable medium in accordance with claim 45, wherein:
said query is formatted in an XPath-type language; and
said document if formatted in an XML-type language.
50. A computer-readable medium in accordance with claim 28, further comprising:
determining if a selected read query is in a read cache, wherein:
a read query is utilized to read a fixed length segment of passing stream data;
if said selected read query is in said read cache, indicating if said selected query is satisfied;
if said selected read query is not in said read cache, add said selected read query to a prefetch set;
obtaining at least one fixed length segment of passing stream data; and
consuming said at least one fixed length segment of passing stream data until all unresolved queries in said prefetch set are resolved.
51. A computer-readable medium in accordance with claim 28, further comprising:
creating a new read cache comprising existing unresolved queries and receiving a source data stream, wherein a read cache is utilized to read a fixed length segment of passing stream data;
creating an empty write cache, wherein a write cache is utilized to provide fixed length segments of stream data;

associating said empty write cache with an existing instance of a re-writer, wherein a re-writer is utilized to provide a copy of a fixed length segment of stream data; and

providing a copy of said source data stream as an output data stream from said empty write cache.

52. A computer-readable medium in accordance with claim 28, further comprising:

obtaining a fixed length segment of source stream data from an associated read cache;

consuming said fixed length segment of source stream data;

resolving pending queries on said source stream data;

creating a mutator stream of data from said source stream of data, wherein mutations in said mutator stream are in accordance with write requests cached in an associated write cache;

consuming an output stream of data from said mutator stream of data; and

resolving queries on said output stream;

discarding resolved queries; and

embedding values associated with said resolved queries in said output stream.

53. A computer-readable medium in accordance with claim 51, further comprising:

determining if a read-query to read said data steam is safe, wherein a safe read-query is a semantically correct query;

if said read-query is safe:

searching a write cache for a value associated with said read-query;

if said value is found in said write cache, returning said value to a user;

if said value is not found in said write cache, forwarding said read-query to a read cache;

if said query is not safe:

obtaining said output stream from a current instance; and

extracting said read-query from said obtained output stream.

54. A computer-readable medium in accordance with claim 51, further comprising:

determining if a write request to write a data stream is safe, wherein a safe write request is semantically correct;

if said write request is safe:

searching a write cache for a value associated with said write request;

if said value is found in said write cache, replacing said associated value with a value of said write request;

if said value is not found in said write cache, creating a new entry in said write cache having said value of said write request;

if said write request is not safe:

creating an intermediate stream writer for separating existing write requests from said unsafe write request;

creating a new stream writer; and

returning said new stream writer to a user.

55. A system for processing a data stream embodying a hierarchically structured document, said system comprising:

a receive pipeline for:

receiving said data stream;

partitioning said data stream into fixed length segments utilizing said hierarchical structure to determine a length of each fixed length segment; processing said fixed length segments in a pipeline fashion; and providing said processed fixed length segments to a storage medium; said storage medium coupled to said receiving pipeline and coupled to a transmit pipeline; and

said transmit pipeline for:

receiving processed data from said storage medium; processing fixed length segments in a pipeline fashion.

56. A system in accordance with claim 55, further comprising:

a first mapping component coupled to said receive pipeline for converting said hierarchically structured document into said data stream; and

a second mapping component coupled to said transmit pipeline for converting a processed data stream into a processed hierarchically structured document.

57. A system in accordance with claim 56, wherein said hierarchically structured document and said processed hierarchically structured document are formatted in accordance with a transport protocol.

58. A system in accordance with claim 55, said receive pipeline comprising:

a decoder for decoding said fixed length segments; and

a parser coupled to said decoder for parsing said decoded fixed length segments.

59. A system in accordance with claim 55, said transmit pipeline comprising:

a serializer for converting data received from said storage medium into fixed length segments; and

an encoder coupled to said serializer for encoding said serialized fixed length segments.

60. A system in accordance with claim 55, wherein:

 said receive pipeline partitions said fixed length segments into fragments; and
 said storage medium stores said fragments therein.

61. A system in accordance with claim 60, wherein said storage medium comprises:

 a database, and

 a database table comprising:

 meta data associated with said document;

 queries over said document and respective results;

 a first fragment of said document; and

 locations within said database and sizes of all fragments of said document other than said first segment.